

Lab 02 – Maven, Tomcat e GitHub

Neste laboratório vamos aprender a utilizar a ferramenta **Maven** para organizar nosso projeto e em seguida, vamos instalar e configurar o servidor web **Apache Tomcat** onde vamos rodar nossas aplicações. Daremos também uma introdução ao GitHub.

Exercícios

Exercício 1: Criar um projeto Maven.

Exercício 2: Configurando arquivos pom.xml e web.xml

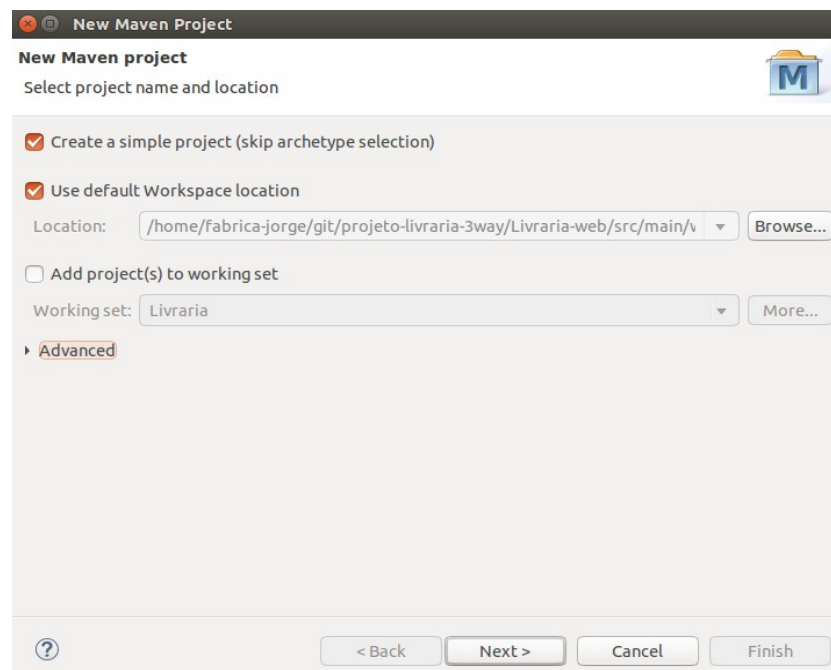
Exercício 3: Iniciando servidor Tomcat

Exercício 4: Utilizando GitHub

Exercício 1 – Criar um projeto Maven

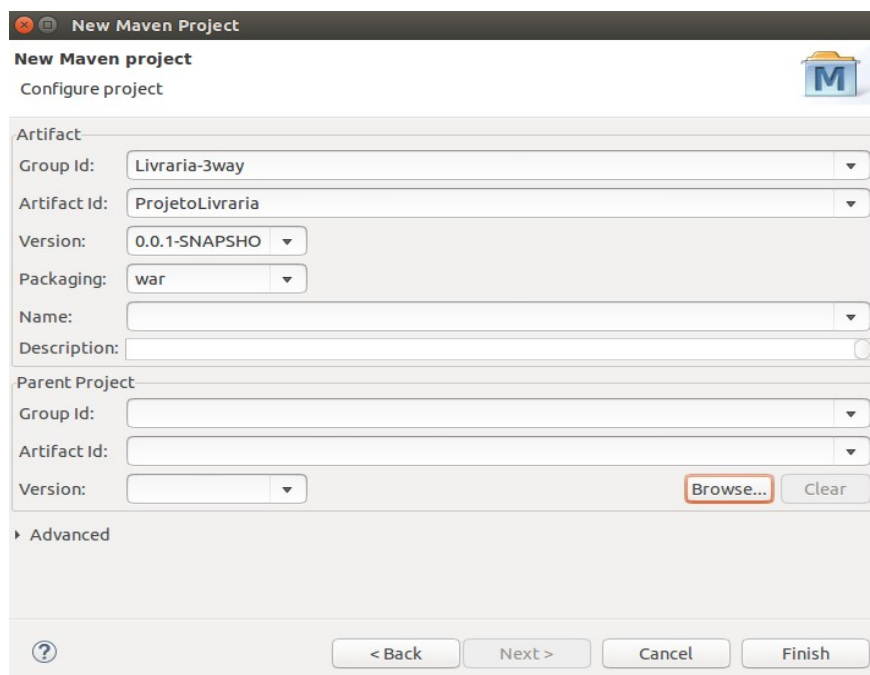
1.1 – Novo Projeto

1. Crie um projeto maven no eclipse (File → New → Maven Project).



Selecione as opções **Create a simple project(skip archetype selection)** e **Use default Workspace location** , assim como na figura e clique em **Next**.

2. Na tela de configuração do projeto, devemos prestar atenção a algumas opções: **Group Id**, **Artifact Id** e **Packaging**



- **Group Id** (um identificador de qual empresa/grupo o projeto pertence): `Livraria-3way`
- **Artifact Id** (nome do Projeto): `ProjetoLivraria`
- **Packaging** (como o projeto será empacotado): `war` (Web Archive)

Concluídas essas alterações, clique em **Finish** para criar o projeto.

Exercício 2 – Configurando arquivos pom.xml e web.xml

1. Depois que nosso projeto foi criado, precisamos fazer alguns ajustes para que ele rode adequadamente no **Eclipse**. Vamos começar pelo **pom.xml**, a versão gerada pelo Eclipse é a seguinte:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <artifactId>Livraria-web</artifactId>
  <packaging>war</packaging>
</project>
```

2. Nele devemos adicionar algumas configurações para que o **maven/eclipse** saiba qual versão do **java** deve usar no projeto:

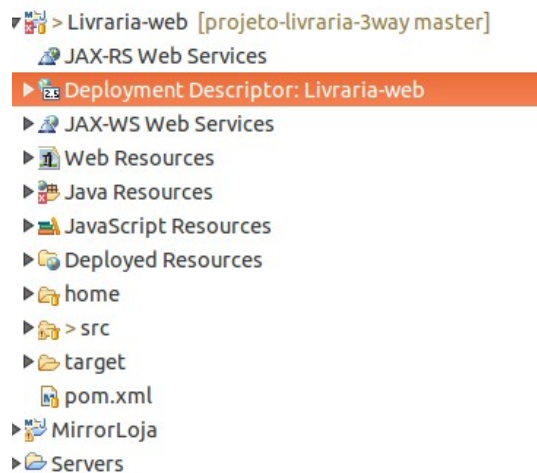
```
<properties>
```

```
<maven.compiler.source>1.7</maven.compiler.source>
<maven.compiler.target>1.7</maven.compiler.target>
</properties>
```

Nesse caso foi utilizado o java 1.7, mas verifique qual a versão usada em seu projeto antes de acrescentar essa tag no seu pom.xml .

3. Agora, atualize seu projeto Maven. Clique com o botão direito do mouse em seu projeto, vá até a opção **Maven** → **Update project**.

4. Feita a atualização do projeto, precisamos criar um arquivo web.xml. Esse arquivo pode ser criado e implementado, porem o maven já nos oferece uma opção de criação desse arquivo com mais facilidade. Em seu projeto, clique com o botão direito do mouse em cima de **Deployment Descriptor** e procure pela opção **Generate Deployment Descriptor Stub**.



Concluindo esse passo, você pode notar que dentro da sua pasta **webapp** foi criado uma pasta **WEB-INF** com o arquivo **web.xml** já criado dentro dessa pasta.

Exercício 3 – Iniciando Servidor Tomcat

3.1 – Baixando Tomcat

1. Vá ao site do Apache Tomcat e baixe o zip da versão mais recente. (<https://tomcat.apache.org/download-80.cgi>)

- Core:
 - [zip \(pgp, md5, sha1\)](#)
 - [tar.gz \(pgp, md5, sha1\)](#)
 - [32-bit Windows zip \(pgp, md5, sha1\)](#)
 - [64-bit Windows zip \(pgp, md5, sha1\)](#)
 - [32-bit/64-bit Windows Service Installer \(pgp, md5, sha1\)](#)

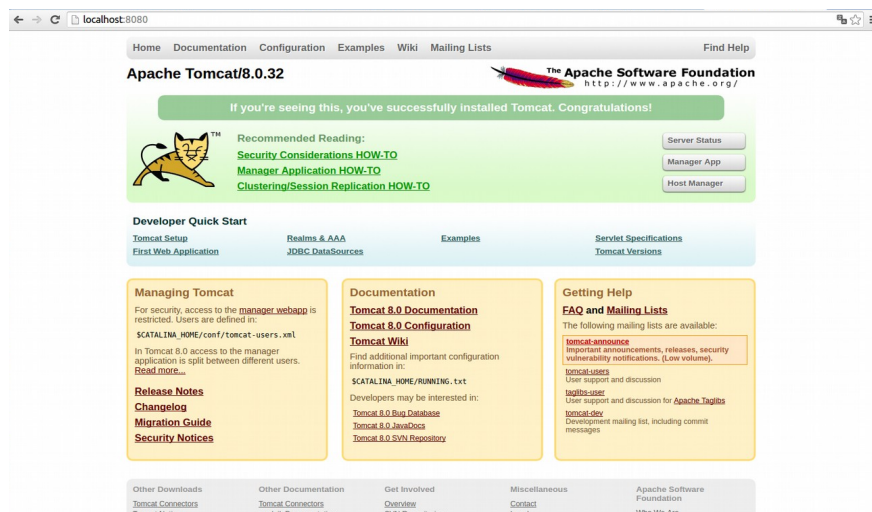
2. Em seguida, descompacte o arquivo baixado em uma pasta com o mesmo nome.

No console, entre na pasta **apache-tomcat-8.0.33** → **bin** e execute o

arquivo startup.sh para iniciar o Tomcat.

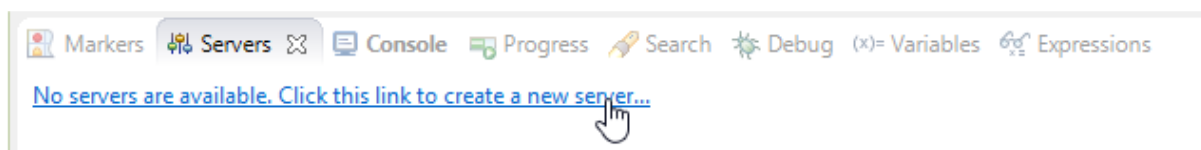
```
fabrica-jorge@fabricajorge:~/Área de Trabalho/apache-tomcat-8.0.32/bin$ sh startup.sh
```

Iniciado o servidor, abra um navegador e entre no endereço **localhost:8080/**. Deverá aparecer uma tela como abaixo. Essa tela é um exemplo que já vem na pasta do tomcat.

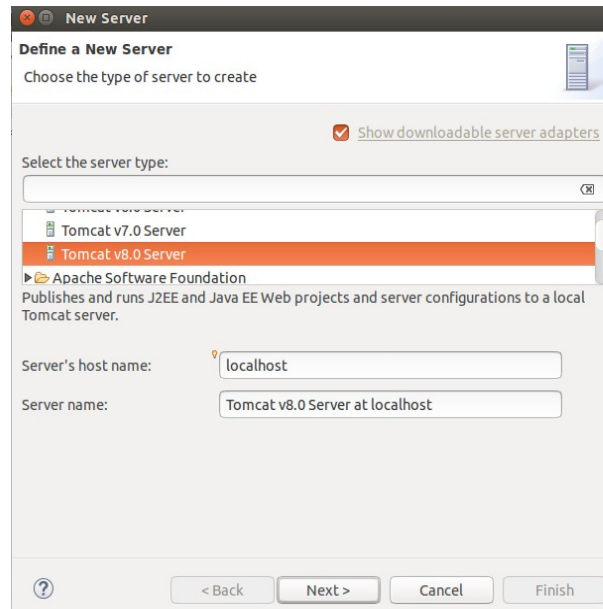


3.2 – Configurando Tomcat no Eclipse

1. No Eclipse JEE, na parte de baixo vão aparecer várias abas. Entre na aba **Servers**. Como ainda não associou o tomcat ao seu projeto, a aba vai estar vazia. Clique no link para adicionar um novo server, assim como a imagem abaixo.

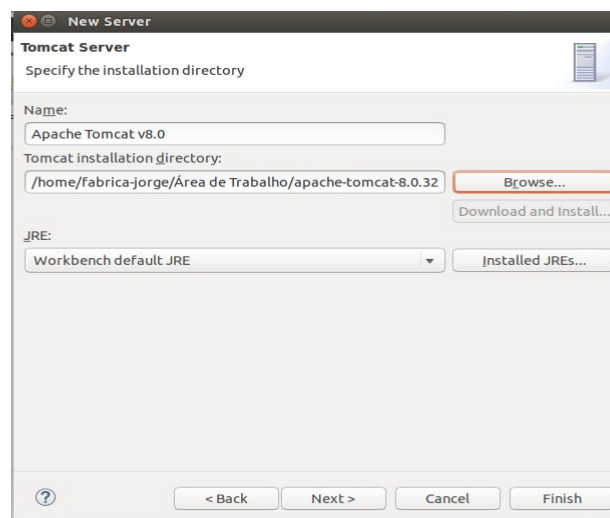


2. Ao clicar no link, aparecerá a tela abaixo, para definir um novo server. Procure pela pasta **Apache Tomcat** e selecione o **Tomcat v8.0 Server**.

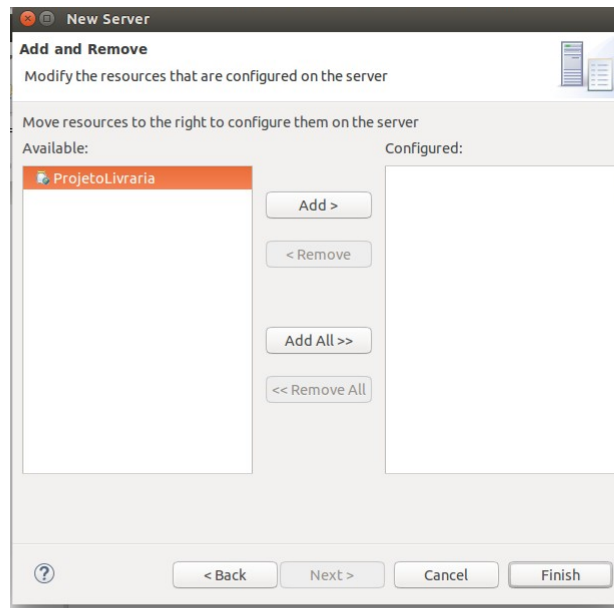


Concluído, clique em **Next>**.

3. Nessa tela, você deve adicionar o local onde a pasta do apache-tomcat está. Clique em **Browse...** e selecione a pasta **apache-tomcat-8.0.33**



4. Agora adicione seu projeto para ser configurado de acordo com o servidor criado. Nessa tela, selecione seu projeto e clique em **Add**. Concluído esse passo, clique em **Finish**.



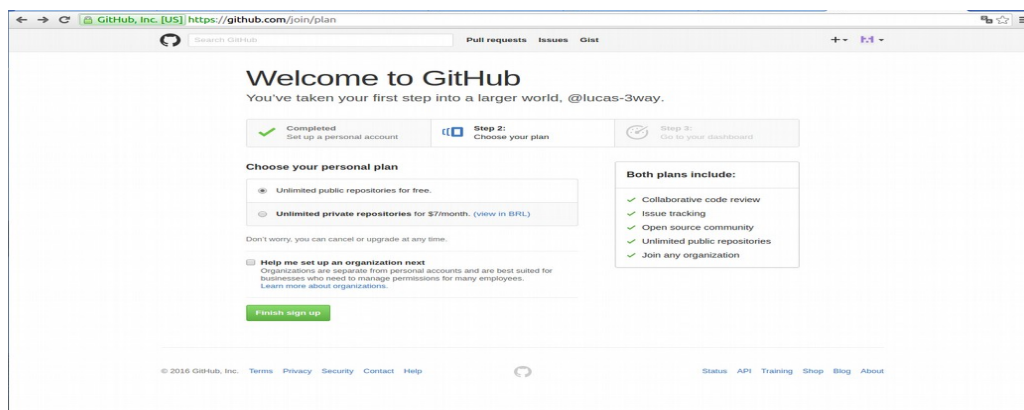
Pronto, suas aplicações já podem funcionar no Tomcat através do endereço **localhost:8080/Nome-do-projeto**.

Exercício 4 – Utilizando GitHub

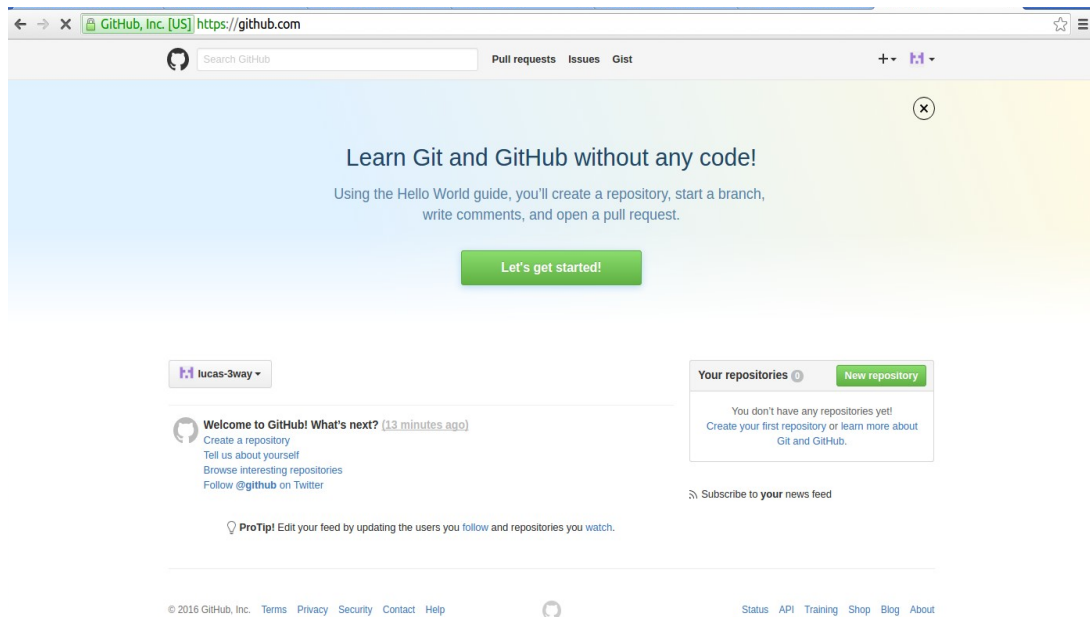
4.1 – Cadastro

1. Vá ao site do GitHub e faça seu cadastro. A partir dele você vai submeter seus projetos na página. (<https://github.com>).

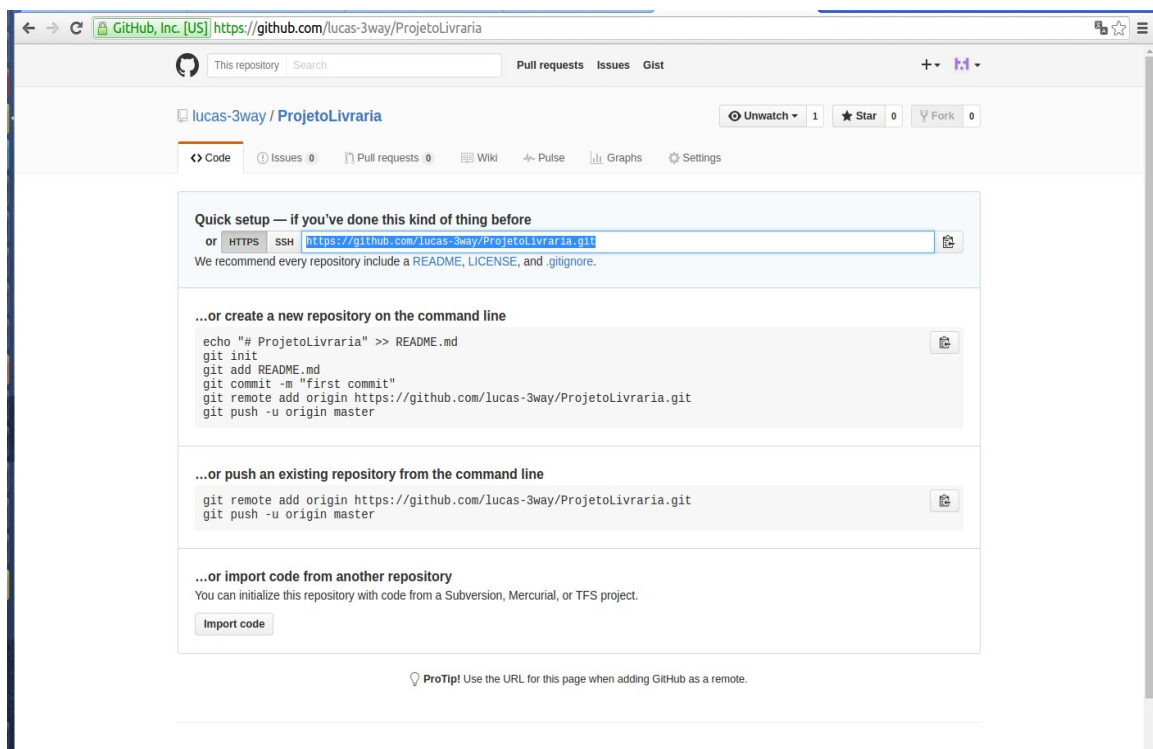
2. Selecione o modelo do plano (**Unlimited public repositories for free. - Recomendado**) e clique em **Finish sign in**.



3. Em sua página inicial, aparece um botão de **Get Started**, onde o site explica as funcionalidades do GitHub.

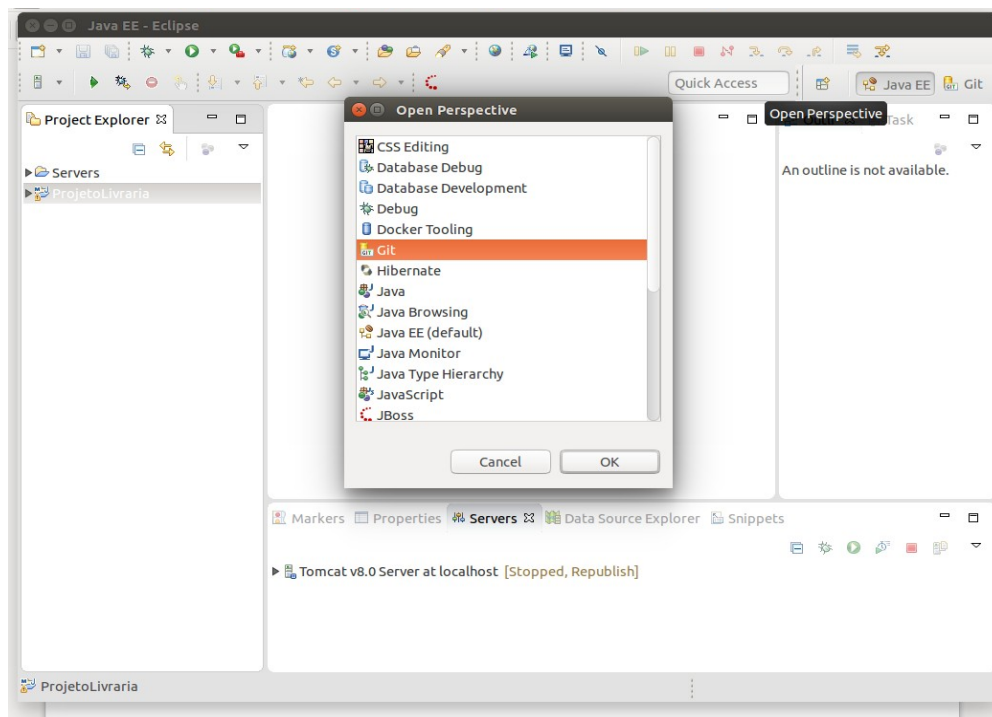


4. Confirme seu endereço de e-mail e clique no botão **New Repository**. Coloque um nome que associe esse repositório criado ao seu projeto. Em seguida, o git irá te mandar um endereço. Copie esse endereço que mais tarde será usado.

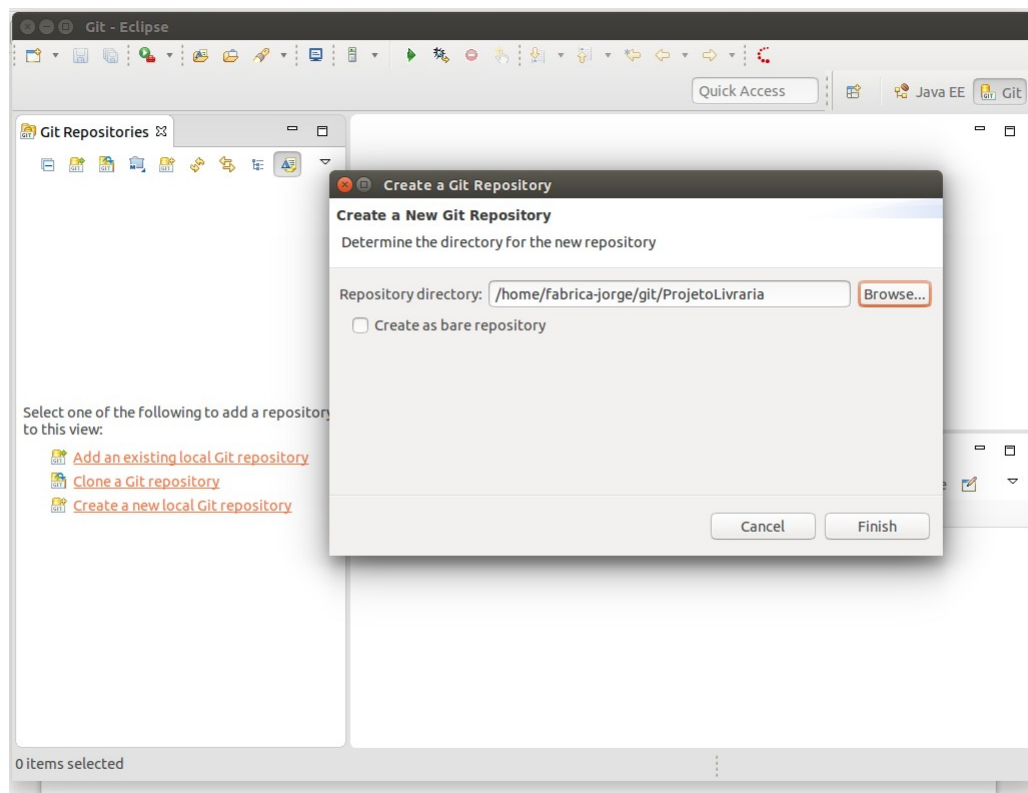


4.2 – GitHub no Eclipse

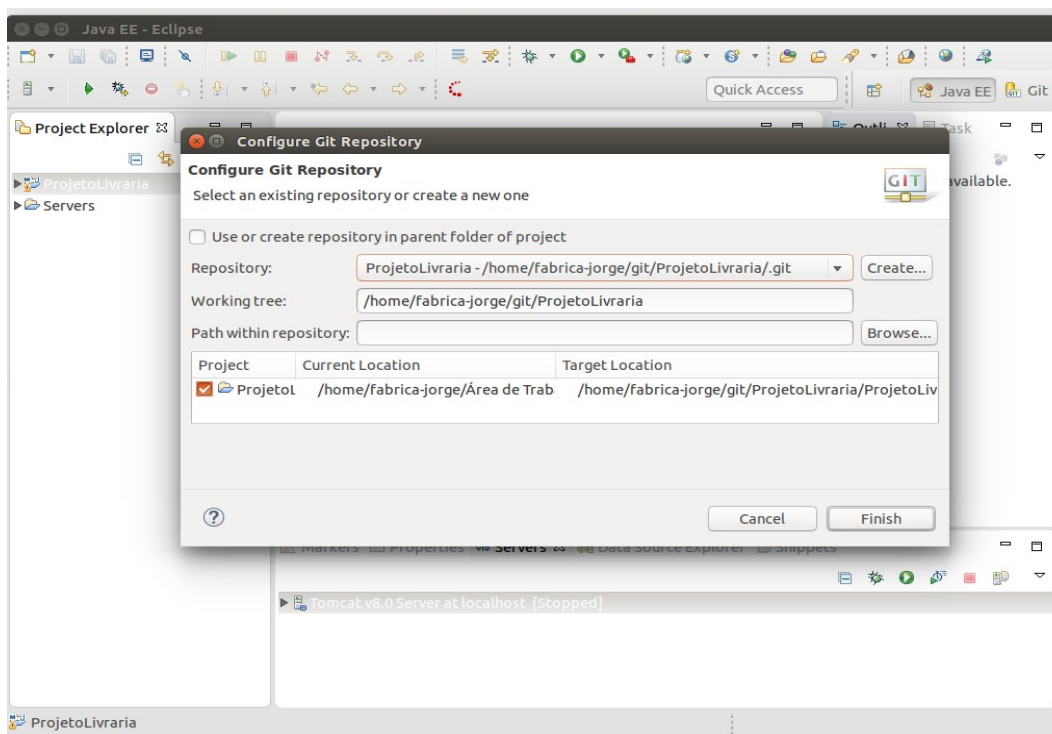
1. No eclipse, já vem o Git, uma integração entre o eclipse e o GitHub. Para usa-lo, abra uma perspectiva **Git**, no canto direito da tela.



2. Nessa perspectiva, você vai criar um novo repositório git. Escolha uma pasta vazia para criar esse repositório, dentro de uma pasta /git para maior organização.



3. Criado o repositório, volte para a perspectiva Java EE e clique com o botão direito em cima do seu projeto. Vá em **Team** → **Share Project**. Abrirá uma tela onde você tem que selecionar qual repositório vai ser usado para compartilhar o projeto. Selecione o repositório que acabou de criar e clique em **Finish**.

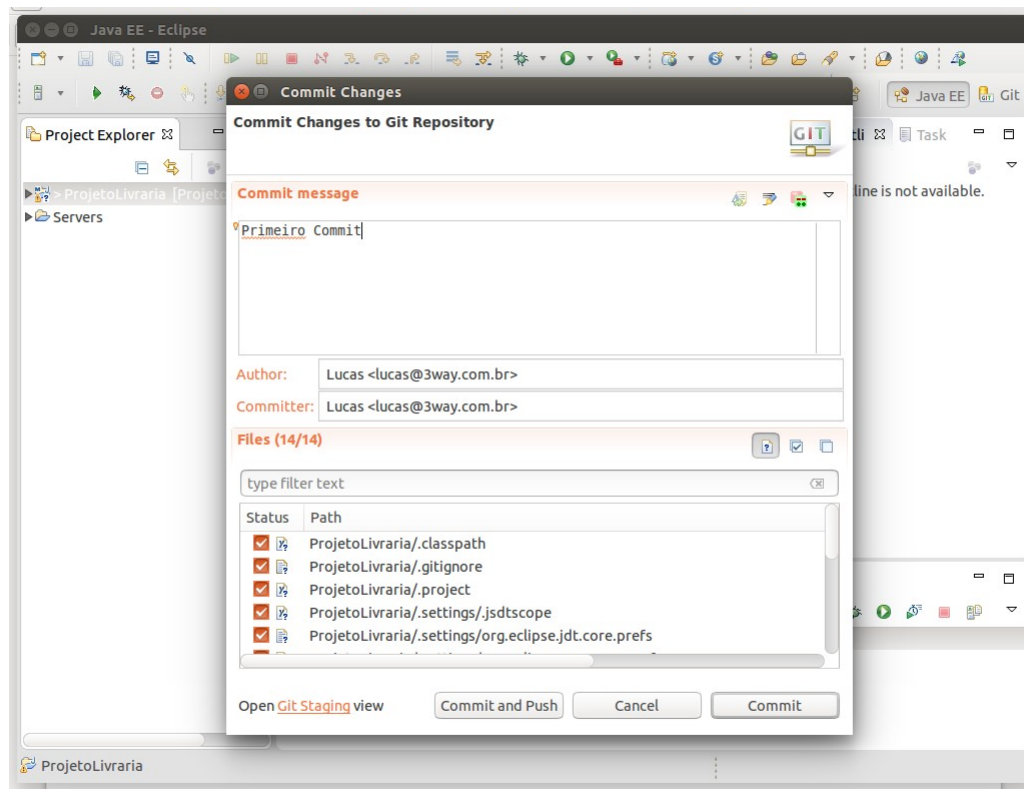


4. Pronto, seu projeto está associado ao GitHub. Vamos aprender agora a organizar e mandar seu projeto para sua página no **github.com** para ser armazenado em nuvem e compartilhado com outros usuários do git. O Eclipse já mostra para nós em que estado se encontra seu projeto. Veja o índice abaixo:

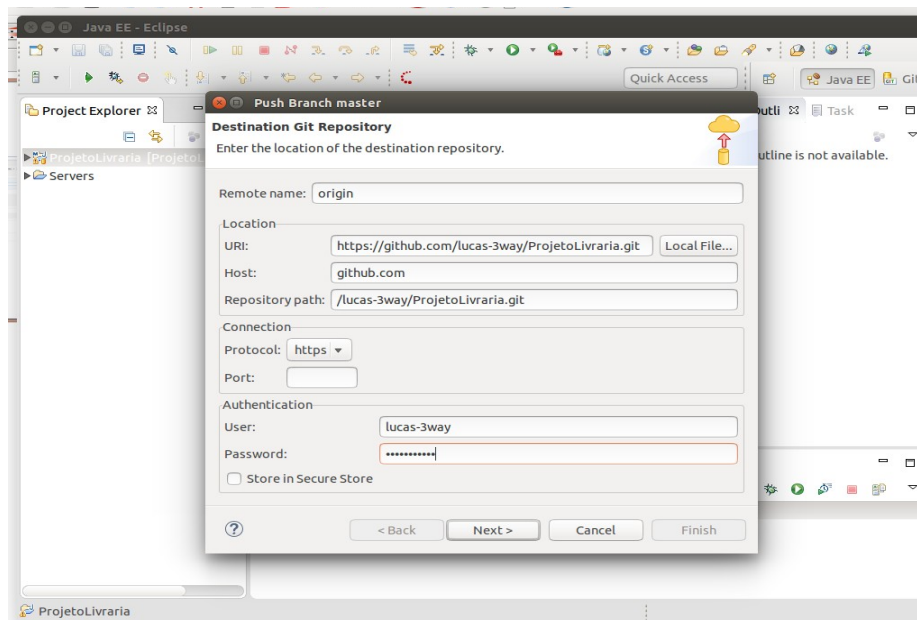
```
> Project [repository|Conflicts master ↑ 2 ↓ 1]
  > folder
    tracked.txt
    untracked.txt
    ignored.txt
    > dirty.txt
    staged.txt
    > partially-staged.txt
    added.txt
    removed.txt
    > conflict.txt
    assume-unchanged.txt
```

Primeiramente, devemos fazer o **commit** do nosso projeto. Clique com o botão direito em cima de seu projeto, vá em **Team** → **Commit...**

Escreva um comentário para identificar seu commit, entre com o nome do autor e e-mail e selecione todos os projetos que gostaria de tornar permanentes as mudanças. Nessa tela temos duas opções, o **Commit** e o **Commit and Push**. Vamos usar só o **Commit** por enquanto.



5. Feito o commit do seu projeto, vamos fazer o **Push**, “empurrá-lo” para o site do GitHub. Vá em **Team** → **Push branch 'master'**. Selecione o repositório que você criou no site github.com, aquele endereço copiado no final do exercício **4.1** e adicione seu login do github. Finalizado, clique em **Next**.



Essa próxima parte não vai ser estudada por agora, então clique em **Next**.

Na tela seguinte, o eclipse vai confirmar seu login e em seguida, clique em **Finish**. Esse passo fará com que seu projeto fique armazenado em sua

página do GitHub. É sempre muito importante fazer **commit & push** para deixar seu projeto atualizado na nuvem.

OBS: Esse projeto Livraria-Web será usado mais para frente, no Workshop da parte de Java Web, onde iremos desenvolver uma livraria virtual. Para os laboratórios e testes, é recomendado criar outro projeto.